Un mécanisme de révocation distribué et adaptatif pour les réseaux pair-à-pair

Thibault Cholez, Isabelle Chrisment et Olivier Festor

MADYNES - INRIA Nancy-Grand Est, France
{thibault.cholez,isabelle.chrisment,olivier.festor}@loria.fr

Résumé Avec le déploiement sans cesse croissant des réseaux pair-à-pair, superviser les comportements malveillants des utilisateurs qui dégradent la qualité et les performances de ces réseaux devient un enjeu majeur. Dans ce papier, nous proposons un mécanisme de révocation complètement distribué et adaptatif basé sur la réputation de chaque pair. L'originalité de notre approche est d'intégrer la révocation au cœur des services proposés par le protocole P2P ce qui permet d'éviter les mécanismes complexes de consensus et de chiffrement passant difficilement à l'échelle. Le premier critère utilisé pour construire la réputation d'un pair est sa contribution au réseau afin de combattre les comportements égoïstes. Les résultats préliminaires montrent que les délais induits par le mécanisme ne devraient pas être ressentis par l'utilisateur et que cette solution est robuste aux attaques.

Mots-clés: réseaux P2P, mécanisme de révocation, mécanisme de réputation, compte distribué, KAD

1 Introduction

Les réseaux pair-à-pair (P2P) ont prouvé leur capacité à rassembler et partager de grandes quantités de ressources grâce à la collaboration individuelle de nombreux pairs. Ils proposent de nombreux avantages par rapport au schéma client-serveur classique : ils passent mieux à l'échelle, tolèrent les pannes et distribuent le coût de l'infrastructure.

Cependant, les réseaux P2P rencontrent plusieurs difficultés dues au nombre important d'utilisateurs malveillants. L'absence d'autorité centrale et le comportement individuel de chaque pair rendent leur gestion difficile. Les pairs malveillants peuvent exercer trois sortes d'actions néfastes : l'attaque ou le non respect du protocole P2P, la diffusion de contenu néfaste (malware, pollution, contenu illégal) et les comportements égoïstes. Plusieurs études ont été menées afin d'estimer l'impact de ces comportements sur le réseau. [1] et [8] ont supervisé Gnutella et constaté la tragédie du bien commun qui est la conséquence des comportements égoïstes : 70% des utilisateurs ne partagent rien et 50% des ressources sont partagées par seulement 1% des utilisateurs. Les auteurs mettent ainsi en évidence les limites d'un système basé sur le volontarisme dans un réseau où chacun est anonyme. Ils sont pessimistes quant à l'avenir du réseau si ce phénomène n'est pas résolu. Le phénomène de pollution a été étudié par [11], il apparaît qu'en moyenne 50% des

fichiers audio partagés sur Kazaa sont pollués et même davantage pour les fichiers récents. Ainsi, les comportements malveillants dégradent la qualité de service offerte par les réseaux P2P.

Dans ce contexte, les réseaux P2P ont besoin de contrôler les comportements de leurs utilisateurs. Pour cela, nous avons conçu un mécanisme de révocation complètement distribué et adaptatif. Notre architecture est adaptée aux réseaux P2P structurés qui fournissent davantage de fonctionnalités que les réseaux non-structurés tout en étant plus efficaces dans leur organisation [3]. La révocation est décidée et adaptée en fonction de la réputation de chaque pair accessible grâce à des comptes distants stockés au sein de la table de hachage distribuée (DHT). Pour l'instant, la réputation évolue en fonction de la contribution d'un pair au réseau, mettant ainsi en évidence les comportements égoïstes.

Cet article est organisé comme suit. La section 2 présente les autres travaux concernant la réputation et la révocation dans les réseaux P2P. Le principe général de notre architecture est décrit dans la section 3 qui inclut le concept des comptes distribués stockant la réputation, l'évolution de celle-ci et le mécanisme de révocation. La révocation est plus précisément décrite et illustrée dans la section 4 en prenant pour exemple le réseau KAD. La section 5 concerne l'évaluation des performances et les questions de sécurité. Enfin, la section 6 conclut et présente les travaux futurs.

2 Travaux relatifs

2.1 Réputation

Gérer la réputation dans un environnement distribué est un réel défi. La grande majorité des systèmes de réputation existants n'ont qu'un point de vue local : chaque pair stocke la réputation de ceux avec lesquels il est entré en relation [10] [6]. Si un tel système présente l'avantage de la simplicité, il est cependant impossible de savoir si un pair est malveillant avant d'être entré en contact avec lui car les avis ne sont pas partagés entre les pairs. Ensuite, cette solution n'est pas adaptée aux grands réseaux P2P publics car la probabilité de rencontrer plusieurs fois le même pair est très faible, rendant les réputations construites localement imprécises, si jamais elles sont utilisées. C'est pourquoi les systèmes de crédits implantés dans des applications comme eMule[1] sont peu efficaces pour lutter contre les utilisateurs égoïstes ; une connaissance locale étant insuffisante pour les détecter (trop peu de pairs connus, trop peu de transactions avec chacun).

Plus récemment, le concept de comptes distribués a été présenté dans PeerMint [7] à des fins comptables mais il peut être utilisé pour construire un système global de gestion de la réputation. L'idée est la suivante : chaque pair dispose d'un compte public (i.e. un ensemble d'informations) stocké sur le réseau P2P. Le stockage d'un compte se fait en le distribuant sur plusieurs pairs grâce à une DHT, fondement des réseaux P2P structurés (Chords, Pastry, Kademlia...). Ce groupe de pairs est

[1] Description du système de crédits d'eMule : `http://www.emule-project.net/home/perl/help.cgi?l=1&rm=show_topic&topic_id=134`

périodiquement mis à jour afin de garder l'information au sein du réseau malgré le départ des pairs ; de plus, la réplication rend le mécanisme plus robuste.

2.2 Révocation

La première manière de révoquer un pair est de mettre en place un contrôle d'accès sans autorité centrale, car contraire à la philosophie P2P. Dans [13], les auteurs présentent et analysent différentes approches pour réaliser ce contrôle. Plusieurs politiques sont possibles : un pair entrant doit réunir l'approbation d'un nombre fixé d'autres pairs, ou d'un nombre proportionnel à la taille du groupe. Ils évaluent ensuite les performances des mécanismes de chiffrement utilisés pour implanter le contrôle d'accès. Il apparaît qu'ils passent difficilement à l'échelle ce qui les rend plus adaptés à de petits réseaux ad-hoc avec des contraintes de sécurité qu'à de grands réseaux P2P publics.

Dans [5] est présentée une manière originale de réaliser une révocation dynamique dans un réseau P2P. Lorsqu'un pair détecte qu'un autre est malveillant, il envoie une note de révocation incluant son identité et celle du pair malveillant à l'ensemble du réseau, considérant sa propre existence comme moins importante que la bonne santé du réseau. Le fait de se suicider en voulant révoquer un pair rend le coût de la révocation très élevé ce qui limite les détournements possibles du mécanisme. Cependant, même si ce mécanisme semble adapté aux réseaux P2P, il présente une limitation sévère. En effet, il ne peut être utilisé que dans un réseau privé appartenant à une même entité et non dans un réseau P2P public où chaque pair a un intérêt individuel rendant la révocation par suicide inenvisageable.

3 Architecture générale

3.1 Comptes distribués

Dans notre système, nous reprenons le principe des comptes distribués car il s'avère être un moyen pertinent pour introduire la réputation dans un réseau P2P. Avec ce système, chaque utilisateur se voit attribuer une note évoluant avec les retours des autres membres, de manière à ce que chaque connaissance soit utile à la communauté.

Ainsi stocké dans la DHT, chaque compte d'utilisateur a une adresse logique qui doit persister entre chaque connexion en plus de l'adresse du pair lui-même. L'application eMule utilise déjà deux identités par pair. La première appelée clientID (ou KadID) est une adresse de 128 bits choisie aléatoirement à la première connexion et constitue l'adresse du pair dans la DHT. La seconde est appelée userID et calculée par une fonction de hachage à partir d'informations de l'ordinateur. Elle est utilisée pour le système de crédit et lui est associé un couple de clés publique/privée afin de s'assurer de l'identité du pair revendiquant un userID. Notre solution associe le compte d'un pair à son userID tel que le présente la figure 1. De cette manière le userID ne sert plus à stocker localement les crédits d'un pair mais donne une entrée de la DHT où stocker son compte public.

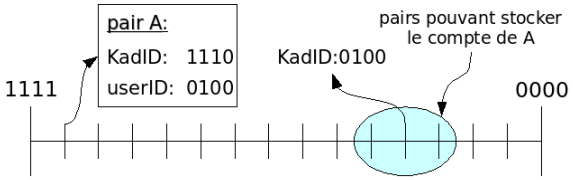

Fig. 1. Stockage d'un compte dans la DHT

3.2 Évolution de la réputation

Dans une application de partage de fichiers, la réputation d'un pair doit croître suivant sa contribution aux ressources du réseau (envoi de données) et inversement décroître avec sa consommation. Ainsi, les utilisateurs sont encouragés à proposer des données intéressant la communauté, des mécanismes existants assurent par ailleurs une diffusion prioritaire des données rares. Avec une telle métrique de réputation, il devient facile d'identifier les utilisateurs égoïstes. La difficulté majeure consiste alors à trouver un moyen fiable de créer et mettre à jour la réputation. Quand un pair arrive sur le réseau, il reçoit une réputation initiale positive permettant les premiers transferts. Cette réputation initiale est nécessaire pour permettre les premières transactions, comme les jetons pour un automate. La réputation n'est pas directement basée sur le Share Ratio mais sur le différentiel d'échange globalement constaté pour un pair.

Ensuite, l'évolution doit être basée sur le fait qu'un transfert implique toujours deux pairs, l'un envoyant et l'autre recevant une quantité de données égale. Pendant un transfert, chaque pair doit indiquer le montant échangé sur une partie de son compte appelée "*blackboard*". Une entrée du *blackboard* affiche ainsi les informations concernant chaque transfert en cours à savoir : l'autre pair intervenant, la direction du transfert et le montant des données transférées. A la fin du transfert, les pairs en charge des comptes ont la responsabilité de mettre à jour la réputation en fonction des informations affichées sur les *blackboards*. Afin d'éviter une entente de pairs malveillants affichant de fausses informations sur le transfert afin d'augmenter leur réputation, les pairs stockant les comptes ne peuvent faire confiance aux informations émises directement par les intéressés. Ils doivent donc communiquer entre eux afin de vérifier la cohérence des annonces faites. Cette condition permet d'assurer qu'un transfert est bien affiché par deux pairs de manière cohérente et évite ainsi les détournements du mécanisme. Tout gain de réputation suite à un transfert a donc inévitablement son corollaire. La figure 2 présente l'utilisation des comptes distribués pendant un transfert et l'évolution de la réputation en conséquence.

La question de vie privée est délicate dans le contexte des réseaux P2P. Le mécanisme de réputation ne constitue pas de nouveau danger pour la vie privée des utilisateurs. En effet, la note de réputation reflète juste la différence entre la quantité de données émises et reçues. Étant donnée une note, il n'est pas possible d'en déduire l'activité d'un pair mais simplement si celle-ci est équilibrée.

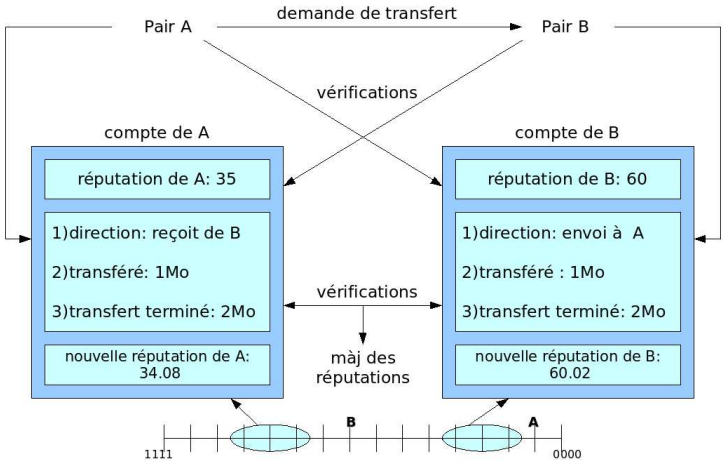

Fig. 2. Utilisation des comptes distribués pendant un transfert

3.3 Le mécanisme de révocation

Le mécanisme de révocation utilise la réputation affichée sur le compte de chaque pair pour décider si, et comment, il doit être révoqué. La réputation peut évoluer jusqu'à un seuil critique qui déclenche la révocation. Comme les réseaux P2P reposent sur les services mutuels fournis entre pairs, un moyen de révoquer de manière totalement distribuée consiste à vérifier la réputation d'un pair avant de lui rendre service. Ainsi, si chaque pair du réseau agit de cette manière, un pair présentant une mauvaise réputation sera automatiquement révoqué, ses requêtes étant rejetées par le réseau. Par ailleurs, ce mécanisme est adaptatif car les services refusés peuvent changer en fonction du critère de réputation utilisé (contribution, qualité du contenu partagé, sécurité...). Les services proposés par les réseaux P2P sont génériques, il y a toujours : une phase d'amorce, un processus de publication (indexation des fichiers partagés sur le réseau), un moteur de recherche, et des connexions directes pour échanger les données.

L'idée d'adapter les sanctions a été présentée par [9] où les auteurs décrivent trois niveaux de contre-mesure possibles en fonction du degré d'égoïsme détecté : décrémenter le TTL des requêtes, ignorer certaines requêtes et déconnecter le pair malveillant. Dans notre approche, chaque service peut être vérifié séparément. Quand un pair présente seulement une mauvaise contribution, il est pertinent de lui enlever le droit de télécharger davantage sans pour autant l'exclure totalement du réseau (en révoquant les autres services) ; de cette manière il pourra faire remonter sa réputation. Au contraire, quand un pair doit être révoqué pour des raisons de sécurité, tous ses droits seront supprimés afin de l'exclure entièrement du réseau. Ceci est résumé par le tableau 1.

	bootstrap et routage	publication et envoi	téléchargement	recherche
Partage	Non	Non	Oui	Non
Sécurité	Oui	Oui	Oui	Non

Tab. 1. Services révoqués en fonction du critère de réputation

4 Application pour le réseau KAD

Cette section explique précisément comment et quels services peuvent être révoqués en prenant exemple sur le réseau KAD. KAD est un réseau P2P populaire proposé par les applications de partage de fichiers eMule et aMule et implantant le protocole P2P Kademlia [12].

4.1 Phase d'amorce

Cette phase est nécessaire avant de rejoindre le réseau. Concrètement, le pair souhaitant rejoindre le réseau demande à quelques pairs connectés de lui envoyer une partie de leurs contacts afin d'initialiser sa table de routage et inversement, pour être référencé par les autres. Une première étape du mécanisme de révocation est donc de vérifier la réputation du pair entrant avant de lui envoyer des contacts. Ce procédé est illustré par la figure 3. Malheureusement, il est envisageable qu'un pair malveillant serve de point d'entrée dans le réseau pour les pairs révoqués en abandonnant cette vérification. Par ailleurs, un pair révoqué peut obtenir indirectement (via le mécanisme de routage itératif) de nouveaux contacts sans demander explicitement le bootstrap. Contrôler la phase d'amorce peut donc potentiellement constituer un premier filtre mais admet plusieurs limitations. Révoquer les autres services ne présente pas ces faiblesses et permet de surcroît d'affiner les sanctions.

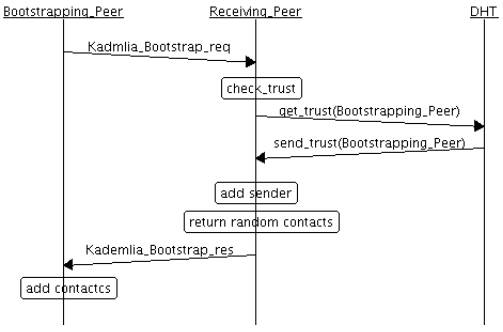

Fig. 3. Vérification de la réputation pendant la phase d'amorce

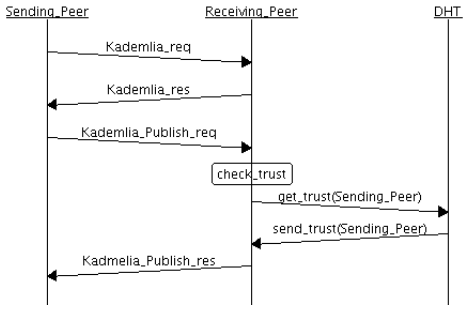

Fig. 4. Vérification de la réputation pendant le processus de publication

4.2 Les autres services

Une fois connecté au réseau, un pair demande des services (publication, recherche, téléchargement...) en envoyant des requêtes aux autres pairs. Dans Kademlia, [12] ceci est réalisé en deux étapes. Dans un premier temps des requêtes *Kademlia_REQ* sont envoyées pour trouver les nœuds capables de fournir le service (selon leur place dans la DHT). Cette phase est un processus itératif essayant de trouver un (ou plusieurs) pair au sein du réseau. La seconde étape est plus intéressante pour le mécanisme de révocation : quand un pair est finalement trouvé, une autre requête spécifique au service demandé est alors envoyée. Vérifier la réputation au niveau de ces requêtes permet de distinguer les services. La figure 4 présente le déroulement d'une demande de publication en y incluant la vérification de la réputation. Les autres services suivent le même principe.

Cependant, il n'est pas pertinent de vérifier la réputation pour le service de recherche de contenu. D'une part, car ce n'est pas un service à travers lequel un utilisateur peut nuire au réseau et d'autre part, la recherche devient inutile quand les autres services sont inactivés en aval. Cela introduirait également une surcharge et des délais supplémentaires pour tous les utilisateurs.

5 Analyse et discussion

5.1 Évaluation des performances

Dans [2] a été menée une évaluation des performances de KAD qui nous permet d'estimer les performances de notre solution. Le délai moyen pour stocker l'information dans le réseau est d'environ 200 secondes. Ce temps est nécessaire pour trouver les 10 pairs (pour la réplication) avec un KadID proche du *hash* de l'information à stocker. Ce délai apparaît à la première connexion d'un pair au réseau si la fonction de Bootstrap est vérifiée. Le délai pour retrouver l'information varie et dépend du niveau de réplication. Plus une donnée est répliquée, plus l'information est robuste et retrouvée rapidement. Les sources maintenant l'information sont trouvées linéairement, le nombre de réponses trouvées étant proportionnel à la durée de la recherche. Un délai de 100 secondes semble suffisant pour rassembler assez de sources pour estimer correctement la réputation d'un pair. Ce délai peut sembler important car il est préliminaire à plusieurs services du réseau. En réalité, 100 secondes supplémentaires pour se connecter sont acceptables, le processus de publication est lui entièrement transparent pour l'utilisateur, et 100 secondes supplémentaires précédant le début d'un téléchargement sont négligeables devant la durée du transfert. Ces éléments montrent que les délais induits par les mécanismes ne devraient pas être ressentis par les utilisateurs. Si toutefois ils étaient trop contraignants, un compromis entre les délais et la fiabilité peut être établit en faisant varier le nombre de réplicats et/ou de réponses attendues.

5.2 Analyse de la sécurité

La sécurité des mécanismes fut une contrainte majeure lors de la conception. Nous avons anticipé les différents comportements malveillants possibles de chaque acteur intervenant afin de rendre le mécanisme plus sûr.

Attaque du mécanisme de réputation : Le premier intérêt d'un pair malveillant est d'empêcher sa réputation de décroître quand il télécharge, ou de l'augmenter davantage que prévu quand il émet. Pour ce faire, un pair malveillant peut tenter de modifier les informations affichées sur le *blackboard* à la fin d'un transfert. Dans le premier cas, le protocole n'autorise pas la diminution d'une quantité affichée sur le *blackboard* car cette action n'a pas de sens lors d'un transfert. Dans le second cas, il y aurait un désaccord entre les *blackboards* des deux intervenants. La réputation doit malgré tout évoluer, sinon le mécanisme serait trop facilement détournable. En cas de désaccord sur le montant du transfert entre les deux pairs, la valeur affichée par le pair téléchargeant est la plus fiable et doit être utilisée. En effet, cette valeur ne peut pas être diminuée et le pair téléchargeant n'a pas intérêt à l'augmenter car sa réputation baisserait d'autant plus. Une entente peut avoir lieu entre deux pairs, mais un seul pourra gagner en réputation au détriment de l'autre, selon le principe des vases communicants.

Il est également possible qu'un pair ayant la charge d'un compte mente quand la réputation lui est demandée. Ce comportement n'a guère d'impact sur les mécanismes car les comptes sont répliqués sur plusieurs pairs. Une demande de réputation recevra donc toujours plusieurs réponses. Comme la majorité des pairs est supposée être honnête, la valeur réelle est la réponse majoritaire. Ceci protège également le mécanisme des pannes byzantines.

La réputation initiale pourrait poser problème si la fonction de hachage générant le userID, donc un nouveau compte, est détournable. Ainsi, un utilisateur malveillant pourrait créer un nouveau compte quand la réputation initiale est consommée, ou transférer la réputation d'un nouveau compte vers son compte principal via de fausses annonces. Une solution possible est décrite dans la partie suivante.

Attaque du mécanisme de révocation : Le mécanisme de révocation est robuste car il est complètement distribué. Si un pair ne respecte pas le protocole en servant un membre révoqué, l'impact sur le réseau sera très limité. La révocation est effective parce que la majorité du réseau décide de refuser de servir les pairs révoqués.

Il existe cependant un moyen de détourner le mécanisme de révocation. Il consiste en une coalition de plusieurs pairs malveillants placés en une même zone de tolérance de la DHT de manière à stocker la majorité des comptes répliqués d'un utilisateur, et ainsi le contôler en mentant collectivement sur sa réputation réelle. La probabilité calculée (2) est basée sur une modélisation du processus d'indexation de KAD qui utilise une zone de tolérance de 8 bits permettant de trouver des pairs capables de stocker une information (en moyenne 4000 pairs pour 1 million d'utilisateurs total). Soit x le nombre de pairs malveillants dans une zone, la probabilité de prendre le contrôle revient à tirer sans remise un certain nombre de pairs malveillants jusqu'à avoir atteint le nombre de réplicats. La probabilité de choisir i pairs malveillants sur 10 réplicats peut donc être modélisée par la loi hypergéométrique de paramètre (10, x, 4000+x) (1), la probabilité de prendre le contrôle revenant à tirer au moins 6 pairs malveillants. Nous pouvons remarquer (courbe 5) que la zone de tolérance dans laquelle une information peut être stockée est assez grande pour rendre très difficile sa prise de contrôle sans d'importants

moyens. Si nous faisons l'hypothèse que la possibilité d'obtenir de nombreux KadID est limitée (par exemple un ID par adresse IP), cette attaque reste peu problable.

L'implantation actuelle de KAD est clairement insuffisante et permet d'annoncer facilement 2^{16} sybils dans une zone [14]. Cependant, plusieurs propositions visent à limiter les *Sybil attack* et permettent d'éviter qu'un pair révoqué puisse facilement revenir sous une autre identité (changement de userID). Dans [14], les auteurs proposent une autorité centrale délivrant les KadIDs par téléphone portable (via SMS). Castro et al. [4] présentent une autre architecture basée sur plusieurs autorités de certification visant à sécuriser l'admission des pairs. Nous pouvons également considérer avec intérêt le projet Keypeer [15] développant une autorité de certification distribuée sur une DHT.

$$P(X = i) = \frac{C_x^i \times C_{4000}^{10-i}}{C_{4000+x}^{10}} \qquad (1)$$

$$P(X \geqslant 6) = \sum_{i=6}^{i \leq 10} P(X = i) \qquad (2)$$

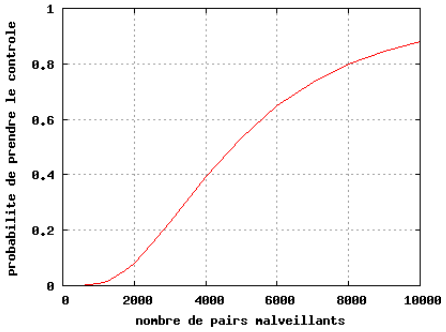

Fig. 5. Probabilité de prendre le contrôle en fonction du nombre de pairs malveillants

6 Conclusion

Superviser les comportements des utilisateurs malveillants est nécessaire pour le bon développement des réseaux P2P publics. Étant donnée la faiblesse des mécanismes d'incitation actuellement implantés et les conséquences négatives des mauvais comportements sur le réseau, il est important de les contraindre à être adaptés à la philosophie P2P. Afin de répondre à ce problème , nous proposons un mécanisme de révocation complètement distribué, adaptatif, et qui ne nécessite pas de consensus ni de moyens de chiffrement complexes. La révocation n'est pas une surcouche du protocole P2P mais est insérée au cœur de ce dernier, contrôlant les services fournis par le réseau. La décision de révoquer est prise selon la réputation du pair qui est fournie par des comptes distribués au sein d'une DHT. Nous avons présenté une première application faisant varier la réputation d'un pair selon sa contribution au réseau. Les premières analyses montrent que les délais induits ne devraient pas être ressentis par les utilisateurs. L'étude recensant les attaques montre que le système est robuste dès lors que les pairs ne peuvent choisir librement leurs identifiants.

Les travaux futurs consistent à implanter ces mécanismes dans KAD afin de vérifier leur fonctionnement et d'évaluer leurs performances (notamment concernant le surcoût généré). Ensuite nous développerons le mécanisme de réputation afin de prendre en considération d'autres critères comme la qualité du contenu partagé pour lutter contre la pollution et la diffusion de malware.

Références

1. Eytan Adar and Bernardo A. Huberman. Free riding on gnutella. *Journal First Monday*, September 2000.
2. Rene Brunner. A performance evaluation of the Kad-protocol. Master's thesis, University of Mannheim and Institut Eurecom, 2006.
3. Miguel Castro, Manuel Costa, and Antony Rowstron. Debunking some myths about structured and unstructured overlays. In *NSDI '05 : Proceedings of the 2nd USENIX Symposium on Networked Systems Design and Implementation*.
4. Miguel Castro, Peter Druschel, Ayalvadi Ganesh, Antony Rowstron, and Dan S. Wallach. Secure routing for structured peer-to-peer overlay networks. *SIGOPS Oper. Syst. Rev.*, 36(SI) :299–314, 2002.
5. Jolyon Clulow and Tyler Moore. Suicide for the common good : a new strategy for credential revocation in self-organizing systems. *SIGOPS Oper. Syst. Rev.*, 40(3) :18–21, 2006.
6. Elizabeth Chang Farookh Khadeer Hussain and Omar Khadeer Hussain. State of the art review of the existing bayesian-network based approaches to trust and reputation computation. In *ICIMP 2007 : The Second International Conference on Internet Monitoring and Protection*, July 2007.
7. David Hausheer and Burkhard Stiller. Peermint : Decentralized and secure accounting for peer-to-peer applications. In Raouf Boutaba, Kevin C. Almeroth, Ramón Puigjaner, Sherman X. Shen, and James P. Black, editors, *NETWORKING*, volume 3462 of *Lecture Notes in Computer Science*, pages 40–52. Springer, 2005.
8. Daniel Hughes, Geoff Coulson, and James Walkerdine. Free riding on gnutella revisited : The bell tolls ? *IEEE Distributed Systems Online*, 6(6) :1, 2005.
9. Murat Karakaya, Ibrahim Korpeoglu, and Ozgur Ulusoy. A distributed and measurement-based framework against free riding in peer-to-peer networks. In *P2P '04 : Proceedings of the Fourth International Conference on Peer-to-Peer Computing*, pages 276–277, Washington, DC, USA, 2004. IEEE Computer Society.
10. Mujtaba Khambatti, Partha Dasgupta, and Kyung Dong Ryu. A role-based trust model for peer-to-peer communities and dynamic coalitions. In *IWIA '04 : Proceedings of the Second IEEE International Information Assurance Workshop*, page 141. IEEE Computer Society, 2004.
11. J. Liang, R. Kumar, Y. Xi, and K. Ross. Pollution in peer-to-peer file sharing systems. In *IEEE Infocom*, pages 1174–1185, march 2005.
12. Petar Maymounkov and David Mazieres. Kademlia : A peer-to-peer information system based on the xor metric. In *IPTPS '01 : Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 53–65. Springer-Verlag, 2002.
13. Nitesh Saxena, Gene Tsudik, and Jeong H. Yi. Admission control in peer-to-peer : design and performance evaluation. In *SASN '03 : Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, pages 104–113. ACM Press, 2003.
14. Moritz Steiner, Taoufik En Najjary, and Ernst W Biersack. Exploiting KAD : possible uses and misuses. *Computer communications review, Volume 37 N 5*, October 2007.
15. Rita H. Wouhaybi and Andrew T. Campbell. Keypeer : A scalable, resilient distributed public-key using chord. columbia university. Technical report, Columbia University, November 2005.